2018 10th International Conference on Cyber Conflict
CyCon X: Maximising Effects
T. Minárik, R. Jakschis, L. Lindström (Eds.)
2018 © NATO CCD COE Publications, Tallinn

Screen Watermarking for Data Theft Investigation and Attribution

David Gugelmann[1]
ETH Zurich
Zurich, Switzerland
david.gugelmann@alumni.ethz.ch

David Sommer[1]
ETH Zurich
Zurich, Switzerland

Vincent Lenders
armasuisse
Thun, Switzerland
vincent.lenders@armasuisse.ch

Markus Happe
ETH Zurich
Zurich, Switzerland
markus.happe@alumni.ethz.ch

Laurent Vanbever
ETH Zurich
Zurich, Switzerland
lvanbever@ethz.ch

Abstract: Organizations not only need to defend their IT systems against external cyber attackers, but also from malicious insiders, that is, agents who have infiltrated an organization or malicious members stealing information for their own profit. In particular, malicious insiders can leak a document by simply opening it and taking pictures of the document displayed on the computer screen with a digital camera. Using a digital camera allows a perpetrator to easily avoid a log trail that results from using traditional communication channels, such as sending the document via email. This makes it difficult to identify and prove the identity of the perpetrator. Even a policy prohibiting the use of any device containing a camera cannot eliminate this threat since tiny cameras can be hidden almost everywhere.

To address this leakage vector, we propose a novel screen watermarking technique that embeds hidden information on computer screens displaying text documents. The watermark is imperceptible during regular use, but can be extracted from pictures of documents shown on the screen, which allows an organization to reconstruct the

1 equally contributing authors

place and time of the data leak from recovered leaked pictures. Our approach takes advantage of the fact that the human eye is less sensitive to small luminance changes than digital cameras. We devise a symbol shape that is invisible to the human eye, but still robust to the image artifacts introduced when taking pictures. We complement this symbol shape with an error correction coding scheme that can handle very high bit error rates and retrieve watermarks from cropped and compressed pictures. We show in an experimental user study that our screen watermarks are not perceivable by humans and analyze the robustness of our watermarks against image modifications.

Keywords: *data theft, investigation, attribution, screen watermarking, malicious insiders, infiltration*

1. INTRODUCTION

Organizations not only need to protect their proprietary information from external attackers but also from insiders [17], i.e., agents infiltrating the organization or malicious employees. To this end, data loss prevention (DLP) solutions are increasingly deployed. State-of-the-art DLP software can either be configured only to log or additionally to block users' actions, such as accessing the Internet, sending emails, printing, taking screenshots or accessing external media. Consequently, data leakage via these conventional communication channels can either be prevented or there is at least a log trail that shows a perpetrator's actions. This log trail can be used as evidence against the malicious insider in forensic investigations. However, DLP systems cannot prevent insiders from taking pictures of a computer screen with a digital camera. Any employee who is authorized to open a particular document on their computer screen can leak the contained information by taking a picture and sharing it with unauthorized parties. Using a camera allows a perpetrator to easily avoid a log trail, as DLP software cannot detect if a document is being photographed. This makes it difficult to identify and prove the identity of the perpetrator based on a recovered leaked picture. Smartphones with cameras have become ubiquitous and new technologies like digital glasses or lenses are gaining momentum, making this data leakage threat difficult to control [23]. Even a policy prohibiting the use of any device containing a camera cannot eliminate this threat since tiny cameras can be hidden almost everywhere.

We introduce a content-agnostic watermarking approach for textual information displayed on computer screens. The watermark is imperceptible during regular use but can be extracted *a posteriori* from pictures of documents shown on the screen.

This enables an organization to reconstruct the place and time of the data leak from recovered leaked pictures, which greatly facilitates the forensic investigation of data breaches involving leaked pictures of screens. Our contributions are:

- an analysis of the data leakage channel computer screen – digital camera (§3);
- a watermarking schema specifically developed and optimized for this leakage channel (§4); and
- a comprehensive evaluation of the suggested watermarking system – including a user study (§5) – and a discussion of attacks against our attribution approach (§6).

2. RELATED WORK

One can distinguish between watermarking solutions for multimedia files and approaches for text documents. Our scenario shows characteristics of both domains. Watermarks need to be imperceptible on screens showing textual contents and must be retained in pictures of the text.

Basic approaches for images simply place watermarks in the least significant bits of individual pixels of an image [20,2,13]. The resulting small color variations are imperceptible to humans, but most smart phone cameras also cannot capture color variations of individual screen pixels, as we found in preliminary experiments. Caronni [5] encodes the watermark by changing the brightness of multiple contiguous pixels, which is similar to our approach. However, his approach requires the original image for extraction of the watermark, while we do not require the original image. Most advanced multimedia watermarking methods operate in a transformed domain, such as an image's frequency spectrum [7,18,19]. This allows them to embed unnoticeable watermarks by introducing slight modifications in the frequency spectrum. This results in noise patterns in the spatial domain. This noise is not noticeable in colorful images but is usually well visible on text documents [12,1]. Therefore, image watermarking approaches operating in a transformed domain are not suitable for the task at hand.

Existing approaches for watermarking of text documents modify the text directly. Jalil et al. [9] distinguish between *image-based, syntactic*, and *semantic* approaches. Image-based approaches [4,3] adapt the typesetting of the text. Syntactic and semantic approaches modify the text itself. They fragment the text into blocks of words or letters, which are then moved or replaced. However, we have to assume that employees can edit documents. In this case, they will probably notice such text modifications. Furthermore, the integration of text-based watermarks is computational-expensive

and can hardly be embedded in real-time. Hence, text modifications are unsuitable for our scenario.

Piec et al. [16] develop a real-time screen watermarking approach for embedding watermarks into screenshots. Screenshots retain colors perfectly and no geometric distortions occur, which allows them to use standard QR codes with their build-in standard error correction for embedding the watermarks. In contrast, we use custom watermark symbols and error correction codes such that our approach not only works for screenshots, but also for pictures of computer screens, in which various image artifacts are present. Kuhn et al. [11] analyzed in their seminal work various approaches to tamper with as well as eavesdrop on information by modifying and analyzing electromagnetic radiation. However, their work analyzes skilled attackers who use hardware to process electromagnetic radiation, while we focus on an attacker using a commodity camera. Petitcolas et al. [14] present criteria for benchmarking watermark approaches and an overview of attacks against watermarks [15].

Printer stenography is related to our approach. For instance, color laser manufacturers encode the date and time a document was printed with tiny yellow dots on print-outs, which cannot be seen unless the print-out is magnified [24]. Recently, it was reported that printer identification code helped to identify the whistleblower Reality Winner in 2017 [25]. Unlike printer stenography, we encode our hidden information on computer screens.

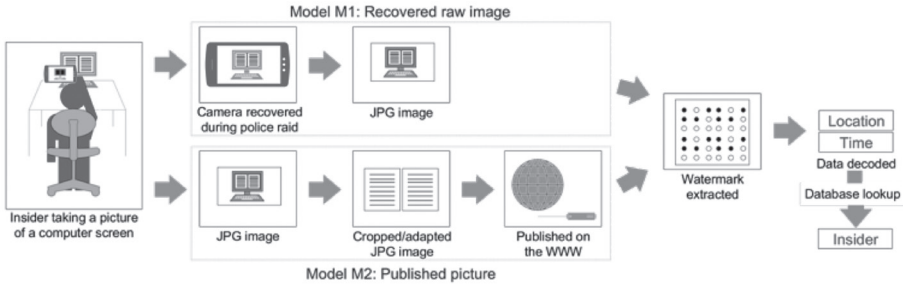
3. PROBLEM STATEMENT AND APPROACH

A. Problem Statement

State-of-the-art security measures cannot prevent insiders from breaching sensitive documents by taking pictures of their computer screens. Taking pictures leaves no log trail that identifies the perpetrator. As a result, it is very difficult to identify the perpetrator based on a recovered leaked picture. We approach this problem with respect to the two scenarios[2] depicted in Figure 1. Both scenarios have in common that: *(i)* an insider (attacker) takes a picture of sensitive information displayed on a screen and *(ii)* a forensic investigator can access the recovered picture and needs to identify the attacker based on the picture. In *scenario M1*, investigators get access to the original, unmodified picture of the camera, e.g., because it was found during a police raid. In *scenario M2*, investigators only see a modified version of the picture as it has been published.

2 Our methodology could also be applied to other scenarios where information needs to be transported in pictures or screenshots.

FIGURE 1. USAGE SCENARIOS: AN INSIDER TAKES A PICTURE OF A COMPUTER SCREEN WHICH IS LATER RECOVERED, EITHER THE ORIGINAL PICTURE (MODEL M1) OR A MODIFIED VERSION OF IT (MODEL M2). THE WATERMARK IS THEN EXTRACTED TO DETERMINE WHEN AND WHERE THE PICTURE WAS TAKEN.



B. Approach

We approach this security threat by embedding hidden watermarks in computer screens. Our watermarks encode information such as the time and the workstation (location). A picture of a watermarked computer screen carries this information. If investigators get access to the (modified) photograph, they can decode this information and identify the perpetrator by verifying who was logged in at the workstation at the time. Watermarking text documents, images, and videos to trace their dissemination is a well-established technique (see §2), but the threat scenario of an insider taking pictures of sensitive data displayed on a screen poses several problems, which make established watermarking techniques unsuitable for this task. In particular:

(a) as the attacker can take a picture at any time, there is no controlled release process and the watermark must be present on any document displayed on the screen at any time;
(b) the watermark must be unnoticeable on text documents, but still be robust against the image artifacts introduced when taking photos of a computer screen; and
(c) the approach should allow for blind extraction, i.e., watermark extraction without the original document.[3]

While traditional text watermarking approaches encode data by modifying individual text passages, we embed information by overlaying a pattern of slightly brighter/darker areas to approach challenge *(a)*. The corresponding overlay mask is independent of the content displayed on the screen and can thus be pre-computed. This makes our watermarking process suitable for real-time embedding. To handle challenge *(b)*, we develop watermarking symbols that are based on the fact that the human eye, especially in light color areas [6], is insensitive to small continuous brightness gradients [2],

3 Alternatively, one would have to record all Desktop interactions resulting in major privacy issues.

while digital cameras capture small changes in brightness well. Further, we modify the design of a traditional convolutional coder and use evolutionary algorithms for deriving optimized generator polynomials in order to handle the high error rates caused by image artifacts. We use redundancy and split our watermarks into sub-watermarks to allow extractions of partly corrupted watermarks. Furthermore, we store cryptographic checksums in our watermarks to allow bit error corrections. To approach challenge *(c)*, we develop an algorithm for blind symbol extraction that is based on the observation that the background color is clearly dominating in typical text documents, allowing us to use local reference brightness values for the symbol decoding.

4. DESIGN

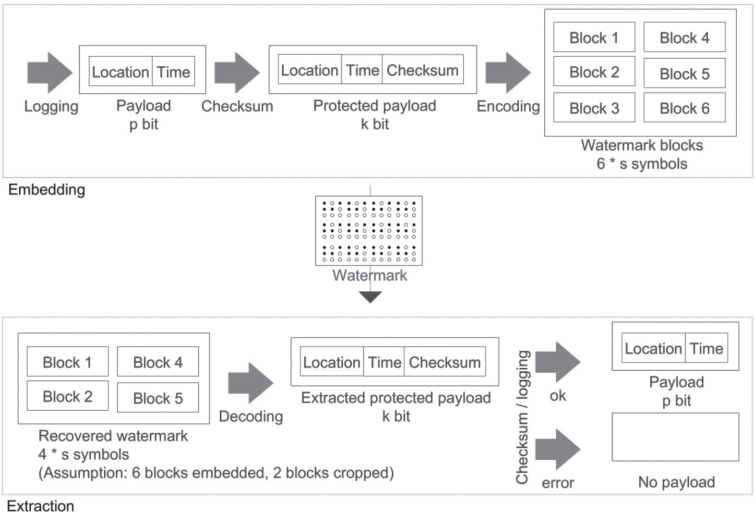
Figure 2 shows the workflow of the proposed watermarking system. The *embedding* process will interact with the graphics card on the watermarked end host (not implemented in the prototype used for this evaluation), while an investigator conducts the *extraction* using standalone software. We assume that a graphic card implementation of the embedding process does not lead to a noticeable increase in CPU usage or power consumption. Even if this assumption does not hold, the user has no baseline for these characteristics that allow them to identify that screen watermarks exist.

Watermark embedding involves the following steps:
(1) Logging: This module creates a bitstring that identifies the end host, user, and a point in time.
(2) Checksum (see §4.C): This module calculates a cryptographic checksum (incorporating a secret user key) for error detection and integrity checking. The checksum block is appended to the payload and the resulting protected payload is provided to the encoder.
(3) Encoding (see §4.B): The protected payload is encoded using an adapted convolutional encoder.
(4) Embedding (see §4.A): Watermark symbols representing the encoded data are generated and placed on the computer's screen.

The extraction of a watermark involves these modules:
(1) Extraction (see §4.A): The watermark symbols are extracted from the recovered picture.
(2) Decoding (see §4.B): The encoded data is decoded using the Viterbi algorithm [21] in order to extract the protected payload.
(3) Checksum/logging (see §4.C): The logging system stores which user was logged in

at the extracted time and location. The corresponding secret user key is retrieved from a database and the cryptographic checksum is verified. If the checksum is correct, the location and time are returned, otherwise the extraction fails with an error.

FIGURE 2. WATERMARKING OF A COMPUTER SCREEN (TOP) AND EXTRACTION FROM A PHOTOGRAPH (BOTTOM). THE PAYLOAD CONSISTS OF *P* BITS. THE CHECKSUM MODULE APPENDS A CHECKSUM TO THE PAYLOAD. THE ENCODING MODULE TRANSFORMS THE PROTECTED PAYLOAD INTO SIX WATERMARK BLOCKS. THIS PROCESS IS REVERSED DURING THE WATERMARK EXTRACTION.



A. Watermarking Symbols for Computer Screens

We introduce watermarking symbols that are a hybrid between traditional text and image watermarking symbols. We operate in the spatial domain, similar to existing text watermarking approaches. This way, the visible artifacts caused by embedding watermarks in a transformed domain are avoided. Still, we avoid the processing-intensive and thus slow text parsing by not changing or moving the text but by overlaying a pattern of slightly brighter and darker areas. Similarly to Caronni [5], we change the brightness of multiple contiguous pixels, which makes our symbols more robust against image artifacts and modifications. However, Caronni's symbol embedding does not allow for blind extraction. To solve this problem, we apply a form of pseudo-differential amplitude modulation. That is, instead of comparing the color values between the watermarked and the original image at the same position in the image, we compare, for each watermark symbol separately, the color within the watermark symbol to the color in the surrounding area. Further, we use circular patterns and soften their shapes by introducing white noise that causes as a smooth gradient

between the watermark center and the surrounding area to avoid sharp contrasts that can become visible on the homogeneous backgrounds in text documents.

The key steps for embedding and extracting watermarks are as follows:

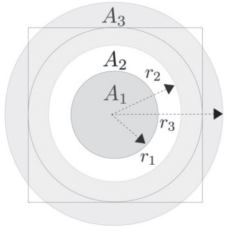
Symbol embedding. Embedding watermark symbols is a two-step process: *(i)* we calculate an overlay mask of slightly brighter/darker areas (symbols) and *(ii)* the watermarking system applies this mask to the screen output. We point out that *(i)* can be pre-computed, thus only *(ii)* is time critical.

Overlay mask. The symbol shape that we use for our approach is shown in Figure 3(a). Every symbol represents one bit. To embed a binary "0", we make the center of the symbol slightly brighter; and to embed a "1", we make the center slightly darker. A watermark consists of a matrix of these symbols. While the brightness of the inner-most circle of the symbol (r_1 in Figure 3(a)) is adapted, a smooth gradient and white noise are applied to the area A_2 to avoid any sharp brightness changes. The watermark decoder compares the background in A_1 to the background in A_3 to tell which binary value the symbol represents. To facilitate the manual extraction of watermarks, the software can further be configured to mark the corners of watermarks using small black markers, which look similar to pixel errors. A photograph of a resulting watermark for an intensity $I_{\max} = 2$ is show in Figure 3(b). Figure 3(c) highlights the watermarking symbols for illustration.

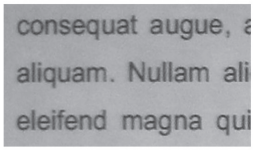
Applying the overlay mask. The application of the overlay mask is quite similar to applying a screen color profile. It requires only local brightness modifications, resulting in a very lightweight embedding process that can be parallelized on a GPU.

Symbol extraction. The extraction of a watermarking symbols from photographs takes place during forensic investigations and is the reverse of the symbol embedding. In contrast to the symbol embedding, this process is not time critical. To extract watermark symbols from a picture, the picture is de-skewed, the watermark symbols are located and the color values of the center of each individual symbol are compared to the surrounding area.

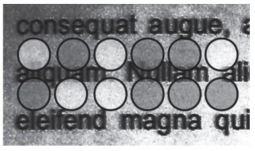
FIGURE 3. A SINGLE WATERMARK SYMBOL (A) AND PICTURES OF A WATERMARKED SCREEN: ORIGINAL (B) AND WITH HIGHLIGHTED SYMBOLS (C).



(a) A single watermark symbol.



(b) Unmodified picture of a watermarked screen.



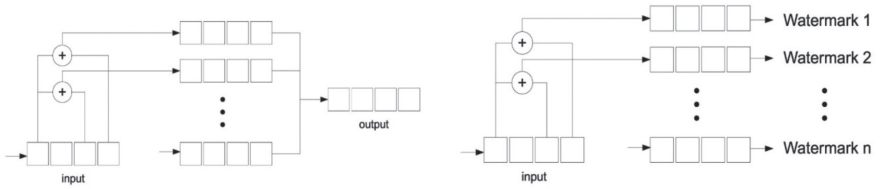
(c) The same picture with watermark symbols highlighted.

B. Encoding of Data in Watermarks

The proposed approach uses error-correcting convolutional codes to achieve a high robustness against incorrectly transmitted symbols. More than one thousand watermark symbols ("physical" bits) fit on a typical screen area of at least 1.024M pixels for our largest symbol size of 32x32 pixels, but we will only need to transport few data bits in a typical setup, therefore we can introduce a high degree of redundancy. Still, this large coding budget is required for a high robustness because error correcting codes for watermarks must be able to operate on short payloads and be robust against various errors [10]. In particular, one has to compensate for cropped images and a very high symbol error rate due to image artifacts. We achieve robustness against cropping by modifying the design of a traditional convolutional encoder and optimizing the error correcting polynomials for short payloads using evolutionary algorithms.

Instead of generating one large watermark, the output of the different generator polynomials used by the encoder is decoupled. This generates multiple smaller, independent sub-watermarks (blocks). Each block carries the complete payload (including a checksum), which allows the decoder to arbitrarily combine the blocks for extracting the payload. That is, on the one hand, one block with few bit errors can already be sufficient to reconstruct the payload. On the other hand, if multiple blocks are available, the decoder can arbitrarily combine these to an optimal combination to compensate for higher bit-error-rate (BER), which leads to very powerful error correcting capabilities. The difference between our and a traditional encoder is illustrated in Figure 4. The traditional encoder merges the output of all generator polynomials to one codeword. In contrast, our design partitions the outputs into smaller sub-watermarks, each with a coding rate of R = 0.5 (termination not included). Combining all sub-watermarks corresponds to the traditional decoder. For decoding the data, we use the common Viterbi algorithm [21].

FIGURE 4. INSTEAD OF MERGING THE OUTPUTS OF THE DIFFERENT GENERATORS (A), WE USE EACH OUTPUT FOR A SEPARATE WATERMARK (B).



(a) Traditional encoder (b) Our encoder

C. Cryptographic Checksum for Error Detection and Integrity Checking

As convolutional codes offer only limited capabilities in detecting errors, a Cyclic Redundancy Check (CRC) [22] is often included in communication protocols to detect decoding errors. We use a cryptographic checksum instead, which additionally allows us to verify the integrity of the extracted message. The checksum block is calculated on the concatenation of the payload and a randomly chosen secret key k_u. Every user u has its own secret key k_u assigned. This protects against accepting a maliciously or accidentally modified message.

5. EVALUATION

We use the following terms throughout the evaluation.

- ***Symbol:*** A symbol is a circular area on the screen that represents one raw bit.
- ***Block:*** As outlined in §4.B, we split a watermark into multiple self-contained blocks. A block is a collection of s symbols.
- ***Symbol size:*** The size of a single symbol (as shown in Figure 3(a)) in pixels.
- ***Watermark intensity:*** the intensity tells how much brighter or darker the symbols are than the surrounding background. We measure the intensity as tuple (Δr, Δg, Δb). The Δ-values are added to or subtracted from the red, green and blue color channel, respectively.

In general, the stronger the watermarks, the more reliable is the watermarking process. But stronger watermarks are also easier to perceive by humans and therefore more disturbing. Thus, we aim to find an operation point at which the watermarks are imperceptible to humans during regular use, but the watermarks can still be reliably extracted from photographs. We evaluate in the following the perceptibility, bit error rates, robustness to image transformation and overall performance of the watermarks.

A. Perceptibility of Embedded Watermarks

1) Setup I

We conduct a user study with 17 adult test subjects working in the defense industry. The aim of the study is to measure and elaborate the visibility of watermarks for different intensities. We embed watermarks of different intensities into a text document; some watermarks are placed in areas with text, while others are placed in a way such that they are not covered by any text. The document is displayed on a Samsung SynchMaster SA450 22 inch screen with a resolution of 1680 x 1050 pixels. The study participants were told that the study was on watermarks, but they did not know what the watermarks looked like. The subjects were asked to read the document. After reading the article, the subjects had to point out which watermarks they could see.

2) Results I

The results of the experiment are presented in Table I. The table distinguishes between watermarks placed on areas where there was no text (background) and watermarks placed in regions with text. All subjects recognized the control watermarks with intensity (20,20,20). But already half of all subjects did not recognize watermarks with intensity (10,10,10) if placed in areas with text. No test subject noted the watermarks of intensity (3,3,3) in text areas. On the other hand, in areas without text, 7 out of 17 subjects spotted watermarks of intensity (3,3,3). The watermarks of intensity (1,1,0) were never identified by any study participant. There is an additional interesting insight not shown in the table. We found that watermarks at the top of the screen were perceived significantly more often than their counterparts at the bottom of the screen. We inspected the screen that was used and found that color contrasts were stronger at the top of the screen than at the bottom.

In summary, we conclude from this study that *(i)* one can use considerably higher intensities for watermarks concealed by text and *(ii)* fine-tuning the intensity of watermarks for different screen regions can be beneficial in order to compensate for the inhomogeneous contrast representation of computer screens.

TABLE I. PERCEPTIBILITY FOR WATERMARKS OF DIFFERENT INTENSITIES. THE PERCEPTION RATE DENOTES THE RATIO OF TEST SUBJECTS IDENTIFYING THE CORRESPONDING WATERMARK.

on white background		in regions with text	
intensity	perception rate	intensity	perception rate
(1,1,0)	0/17	(3,3,3)	0/17
(1,2,1)	5/17	(5,5,5)	4/17
(2,2,2)	5/17	(10,10,10)	9/17
(3,3,3)	7/17	(20,20,20)	17/17

B. Bit error rates

1) Setup II

We measure the bit error rate (BER), i.e., the ratio of symbols that are incorrectly extracted, for different hardware devices. We focus on watermarks that are located in areas with text. We embed watermarks of intensity (2,2,2) in a text document with font size 10pt, display the text document on a Lenovo T430s such that the watermarks cover the whole screen, and we take pictures with different cameras. This laptop features a Twisted Nematic (TN) panel with a resolution of 1600 × 900 pixel. We use a different device for this experiment than for the user study. However, we compared the low contrast characteristics of the panels and found them to be very similar.[4] We measure the BER for three different symbol sizes and four smartphone cameras: Lumia920, SonySk17i, SamsungNexus, and MotorolaXT910. We place two (three for symbol size 20×20) randomly generated watermarks in the document such that they cover the whole screen and take five pictures with each configuration.

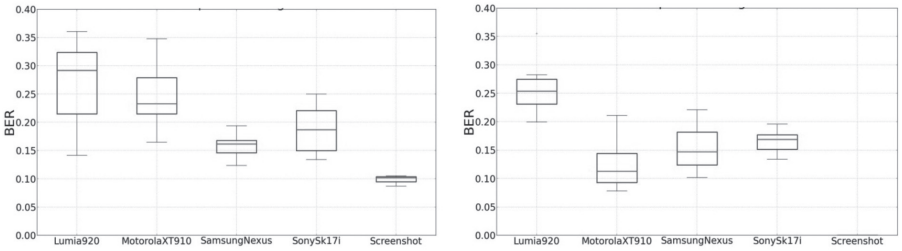
2) Results II

The results for the smallest and largest symbol sizes are shown in Figure 5. Each data point represents the BER for a single watermark. For reference, the right column in each figure shows the BER for symbols directly extracted from a screenshot. We use a screenshot for comparison to measure the influence of the image artifacts caused by taking photographs of the screen. There is a clear trend towards lower BER for larger symbol sizes. This is because pixels representing text are filtered during symbol extraction and more pixels remain after filtering for larger symbols, making the approach more robust. The screenshots also show some bit errors for the two smaller symbol sizes. We confirm this finding by measurements conducted on a watermarked document without any text (not shown in the Figure). For a blank document, the photographs of all symbol sizes achieve a BER of around 0.05 and the screenshots do not exhibit any errors. The user study already showed that contrasts were stronger on the top than on the bottom of the screen. We verified this finding by analyzing the topology of bit errors in Figure 6(a). Indeed, the BER is lower at the top of the screen than at the bottom.

In summary, we conclude that larger symbols are better for watermarks in text areas. For a symbol size of 32×32, we achieve a median BER between 0.12 and 0.25, the maximum BER is 0.28.

4 We tested three different TN panels and one PVA panel. The low contrast characteristics of all these devices were similar.

FIGURE 5. BIT ERROR RATES (BER) FOR WATERMARKS IN TEXT AREAS FOR THE SMALLEST AND LARGEST EVALUATED SYMBOL SIZES AND FOUR CAMERAS, AS WELL AS A REGULAR SCREENSHOT FOR COMPARISON. THE BER FOR A SCREENSHOT IS ZERO FOR 32 × 32 SYMBOLS. FIVE PHOTOS HAVE BEEN TAKEN FOR EACH CONFIGURATION.



(a) Symbol size 20 × 20.

(b) Symbol size 32 × 32.

C. Robustness to Image Transformations

We evaluate in the following the robustness of our approach to image transformations in regard to scaling and color adjustments.

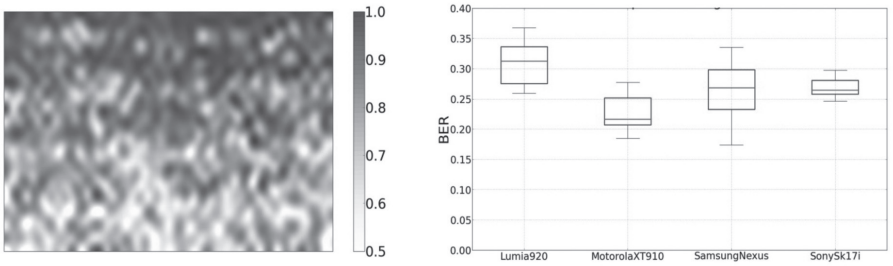
1) Setup III

To simulate a scenario in which images are compressed before being leaked, in this experiment we compress the pictures taken with the mobile phones by a factor of four. This means that the width and height of each image is halved. The resulting pixels are interpolated. A reduction by a factor of four can be considered as a worst-case scenario with respect to image compression for the pictures analyzed in this work, because further decreasing the resolution would make the text in the document very hard to read. Thus, it is unlikely that an attacker would further compress the images.

2) Results III

The resulting BER for a symbol size of 32 × 32 are shown in Figure 6(b). Resizing the images increases the BER by 10 to 15 percentage points compared to their original images, resulting in an average BER of approximately 25%.

FIGURE 6. INHOMOGENEOUS ERROR DISTRIBUTIONS DUE TO DISPLAY CHARACTERISTICS (LEFT) AND BER FOR RESIZED IMAGES (RIGHT).



(a) Ratio of correctly extracted symbols in text areas on a Lenovo T430s.

(b) BER for resized images of watermarked text. Symbol size 32 × 32.

D. Overall Performance

1) Setup V

We calculated the BER for different scenarios in the previous subsections of the evaluation. As the last step of the evaluation, we now relate the BER and the percentage of the available watermarked area to the probability that the transported data can be successfully extracted from a watermark. For this evaluation, we assume that a watermark capacity of $p = 40$ bit is required to encode a user identifier and a timestamp; a payload of $p = 40$ bits results in a protected payload of length $k = 72$ bits and $s = (k + m - 1) * n = 172$ symbols per block (see Figure 2). The parameter $m=15$ represents the length of the used shift register for the convolutional encoder and $n=2$ represents the number of output bits per input bit. We measure the performance of the applied convolutional coding by conducting a Monte Carlo Simulation with 6000 runs. Bit errors are modeled as i.i.d according to the given BER.

2) Results V

The results are shown in Figure 7(b). Every line in this Figure shows the performance of our approach for a different average BER. To give an example, the blue triangle in the upper center of the plot shows that for a BER of 0.25 and 3 recovered watermark blocks, the probability that the data can be successfully extracted from a watermark is around 85%.

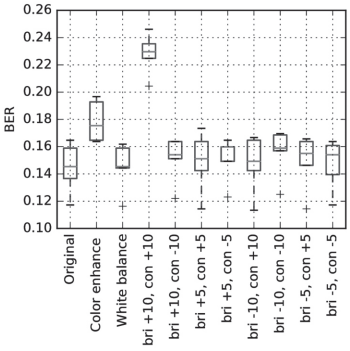
We first focus on pictures without color modifications (raw images). As shown in Figure 5(b), the BER for a symbol size of 32 x 32 is always below 20% for three out of the four mobile devices. Putting this number into Figure 7(b), we see that three out of six watermark blocks are sufficient to decode the data in this case. For the Lumia920, the average BER is 25%, thus we need four to six watermark blocks to

successfully extract the data with high likelihood. The maximum observed BER is 28%. The probability that the payload can be successfully extracted for this case is at least 98%, as Figure 7(b) shows.

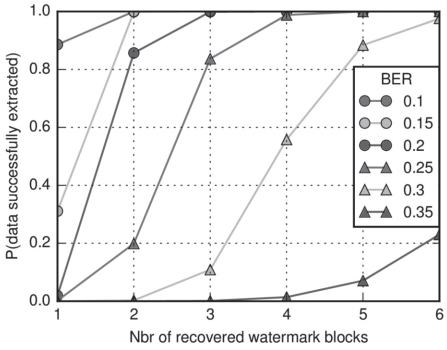
Resizing pictures results in a BER of around 30% (see Figure 6(b)). Figure 7(b) shows that the data can be extracted with a probability of 98% for a BER of 30%. Contrast and brightness changes and automatic color enhancements resulted in a BER below 25% (see Figure 7(a)). Already four out of six watermark blocks are sufficient to reconstruct the embedded data in 99% of cases.

We conclude that we can recover the watermarks from unmodified photographs for all tested smartphones. The Lumia920 introduces a bit error rate of 25%, which reduces the robustness to image modifications, such that 2/3 of the watermark blocks of cropped images are required. For the other three smartphones, we can scale down the image by a factor of four or increase the contrast and brightness by 10% and still extract the encoded data. Watermarked pictures taken with these smartphones are also very robust to cropping of the raw image, only 50% of the watermark blocks are required to extract the watermark.

FIGURE 7. ERROR RATES FOR MODIFIED PICTURES (LEFT) AND OVERALL PERFORMANCE (RIGHT).



(a) The first column shows the BER on the original pictures. The other columns show the resulting BER after applying GIMP's automatic color enhancement, GIMP's white balance function and various brightness (bri) and contrast (con) changes.



(b) Overall performance. The y-axis shows the probability that the data encoded in a watermark can be extracted depending on the number of available watermark blocks and the Bit Error Rate (BER).

6. DISCUSSION

An attacker who is aware of the fact that computer screens are watermarked could try to use our watermarking approach to hold another employee liable for a leaked picture. First, an attacker could attempt to create a fake watermark that contains the identifier of an employee E. However, the attacker also needs to generate the correct cryptographic checksum, which is based on a secret key k_u, otherwise the watermarking system rejects the watermark (see §4.C). An attacker does not know k_u, so they can only guess what the correct checksum is. The odds for guessing the correct checksum is in the order of one in one billion for a 32-bit checksum and six embedded watermark blocks.

Second, an insider could use an unlocked workstation to access the critical information or even access the information with stolen credentials. To detect such a case, one could combine our watermarking approach with biometric techniques that identify the employee currently using a workstation [8].

Third, an insider could take a picture of a document while another employee views the document on their screen. To investigate such and similar cases one would need to complement our approach with CCTV cameras monitoring the office environment. After extracting time and location from a watermark, an investigator could check the surveillance camera recordings of the corresponding office.

Finally, in order to frame an employee *E*, a skilled attacker could take a picture of *E*'s screen, extract the watermark from the picture, and embed it into a picture showing a document that *E* is not supposed to access. The watermark would show where and when the attacker took the picture. This information can be compared against the logs generated by our logging module (see §4), which would show that *E* never accessed the document. Further, CCTV cameras could identify the attacker.

7. CONCLUSION

In conclusion, our proposed watermarking scheme applies imperceptible low-intensity watermarks to the screen. The information embedded with our technique can later be retrieved from photographs or screenshots. We develop a coding scheme based on convolutional codes, which complements the watermarking technique and can cope with the particular challenges of screen watermarking, such as high error rates, inhomogeneous error distributions (caused by the underlying hardware) and partial pictures of screens. We conduct a user study showing that our watermarks are imperceptible during regular use and demonstrate in various experiments that our

watermarks are robust regarding resizing and basic image manipulations. In future work, we will investigate possible attacks against screen watermarks, e.g. by taking advantage of physical screen characteristics, and corresponding protection methods.

ACKNOWLEDGMENT

This work was partially supported by the Zurich Information Security Center. It represents the views of the authors.

REFERENCES

[1.] A. M. Alattar and O. M. Alattar. Watermarking electronic text documents containing justified paragraphs and irregular line spacing. *Proceedings of SPIE* - Volume 5306, Security, Steganography, and Watermarking of Multimedia Contents VI, pages 685-695, Jan 2004.

[2.] W. Bender, D. Gruhl, N. Morimoto, and A. Lu. Techniques for data hiding. *IBM Syst. J.*, 35(3-4):313–336, Sept. 1996.

[3.] A. K. Bhattacharjya and H. Ancin. Data embedding in text for a copier system. In *Proc. Int. Conf. Image Processing (ICIP 99)*, volume 2, pages 245–249. IEEE, 1999.

[4.] J. Brassil, S. Low, N. Maxemchuk, and L. O'Gorman. Electronic marking and identification techniques to discourage document copying. *IEEE J. on Selected Areas in Comm.*, 13(8):1495–1504, Oct 1995.

[5.] G. Caronni. Assuring ownership rights for digital images. In *Verlaessliche ITSysteme*, DUD-Fachbeitraege, pages 251–263. Vieweg+Teubner Verlag, 1995.

[6.] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker. *Digital Watermarking and Steganography*. Morgan Kaufmann, 2007.

[7.] I. J. Cox, J. Kilian, F. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Trans. on Image Processing*, 6(12):1673–1687, Dec 1997.

[8.] S. Eberz, K. B. Rasmussen, V. Lenders, and I. Martinovic. Preventing lunchtime attacks: Fighting insider threats with eye movement biometrics. In *NDSS*, 2015.

[9.] Z. Jalil and A. Mirza. A review of digital watermarking techniques for text documents. In *Proc. Int. Conf. on Information and Multimedia Technology (ICIMT)*, pages 230–234, Dec 2009.

[10.] S. Katzenbeisser and F. A. Petitcolas, editors. *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House, Inc., 2000.

[11.] M. G. Kuhn and R. J. Anderson. *Soft Tempest: Hidden Data Transmission Using Electromagnetic Emanations*, pages 124–142. Springer Berlin Heidelberg, 1998.

[12.] Y. Liu, J. Mant, E. Wong, and S. H. Low. Marking and detection of text documents using transform-domain techniques. *Proceedings of SPIE* - Volume 3657, Electronic Imaging Conference on Security and Watermarking of Multimedia Contents, pages 317-328, 1999.

[13.] N. Nikolaidis and I. Pitas. Robust image watermarking in the spatial domain. *Signal Processing*, 66(3):385 – 403, 1998.

[14.] F. A. Petitcolas. Watermarking schemes evaluation. *Signal Processing Magazine, IEEE*, 17(5):58–64, Sep 2000.

[15.] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn. Attacks on copyright marking systems. In *Proc. Int. Workshop on Information Hiding*, pages 218–238, London, UK, 1998. Springer.

[16.] M. Piec and A. Rauber. Real-time screen watermarking using overlaying layer. In *Proc. Ninth International Conference on Availability, Reliability and Security*, ARES '14, pages 561–570. IEEE Computer Society, 2014.

[17.] Ponemon Institute LLC. 2015 Cost of Data Breach Study: Global Analysis. http://www-03.ibm.com/security/data-breach/, May 2015.

[18.] C.-S. Shieh, H.-C. Huang, F.-H. Wang, and J.-S. Pan. Genetic watermarking based on transform-domain techniques. *Pattern Recognition*, 37(3):555 – 565, 2004.

[19.] T. K. Tsui, X.-P. Zhang, and D. Androutsos. Color image watermarking using multidimensional fourier transforms. *IEEE Trans. on Information Forensics and Security*, 3(1):16–28, March 2008.

[20.] R. Van Schyndel, A. Tirkel, and C. Osborne. A digital watermark. In *IEEE Int. Conf. on Image Processing (ICIP)*, volume 2, pages 86–90 vol.2, Nov 1994.
[21.] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. on Information Theory*, 13(2):260–269, 1967.
[22.] R. Wang, W. Zhao, and G. Giannakis. Crc-assisted error correction in a convolutionally coded system. *IEEE Trans. on Comm*., 56(11):1807–1815, 2008.
[23.] D. Lohrmann. Two New Insider Threats to Consider. *CSO Online* (2013-06-23), https://www.csoonline.com/article/2137207/infosec-staffing/two-new-insider-threats-to-consider.html.
[24.] S. Schoen. Secret Code in Color Printers Lets Government Track You. *Electronic Frontier Foundation Press Release* (2005-10-16). https://www.eff.org/de/press/archives/2005/10/16.
[25.] C. Baraniuk. Why Printers Add Secret Tracking Dots. *BBC, InDepth, Technology* (2017-06-07). http://www.bbc.com/future/story/20170607-why-printers-add-secret-tracking-dots.